Chapter 5

NoSQL and Cloud Paradigm: Characteristics and Classification, Data Storage Technology, and Algorithms

Introduction

- RDBMS have limitations including
  - handling globally distributed data
  - which are semistructured in nature, available in real time, and voluminous.
  - limited processing power and storage capacity.
- The main features of NoSQL include higher scalability, adequately optimized for distributed computing, elastic schemas, etc.
- NoSQL databases solve these challenging issues with various data model, storage structure, and access patterns.

Introduction (2)

- NoSQL supports nonrelational data model and is able to handle a wide variety of data structures. Information can be stored without schema.
- This feature is highly compatible with cloud computing technology where there is a requirement to support heterogeneous kind of data.
- In cloud computing world, data are stored across the globe in various servers, and their size is growing rapidly.
- In this scenario, maintaining the relationship across the data set is a big challenge but is mandatory for all relational databases.

Introduction (3)

- NoSQL can be the one-stop solution as it does not require building the relationship between their records.
- It automatically increases the performance as it reduces the query time, which is done by joining tables containing different data structures.
- NoSQL is also highly distributable in nature, which can store and process information present in various servers, and a cluster of servers can be used to hold a single database.

3.1 Task Scheduling in Cloud

- Execution models in cloud are centered on task execution.
- Task in Cloud activity is expected to be done using underlying resources.
- The key to this execution is to use optimal resources with minimum execution time
- Since there will be several tasks that need to be handled by cloud resources at a time, it becomes critical to schedule tasks efficiently.
- Task scheduling is grouped under two types: *static* and *dynamic* scheduling.
  - Static scheduling refers to scheduling of tasks where information regarding complete sequence of tasks, execution times, and resource mapping is known before execution.
  - Dynamic scheduling, on the other hand, requires system current states of environment and computing machines to make scheduling decision.
- Virtualization techniques are deployed in cloud to map computing resources to run scheduling algorithms.
- Virtual Machines (VMs) provide abstraction of computing and storage resources made available to scheduling algorithms to execute tasks.
- These algorithms are usually managed by a broker that coordinates with virtualization layer to manage VMs

3.1 Task Scheduling in Cloud

3.1.1 List Scheduling

- The fundamental tenet of list scheduling is organizing a task list through a set of policies involving priorities.
- These priorities are governed by the algorithm approach that manages these lists. The execution of tasks is in the order of the highest-priority task, assigning a resource (VM).

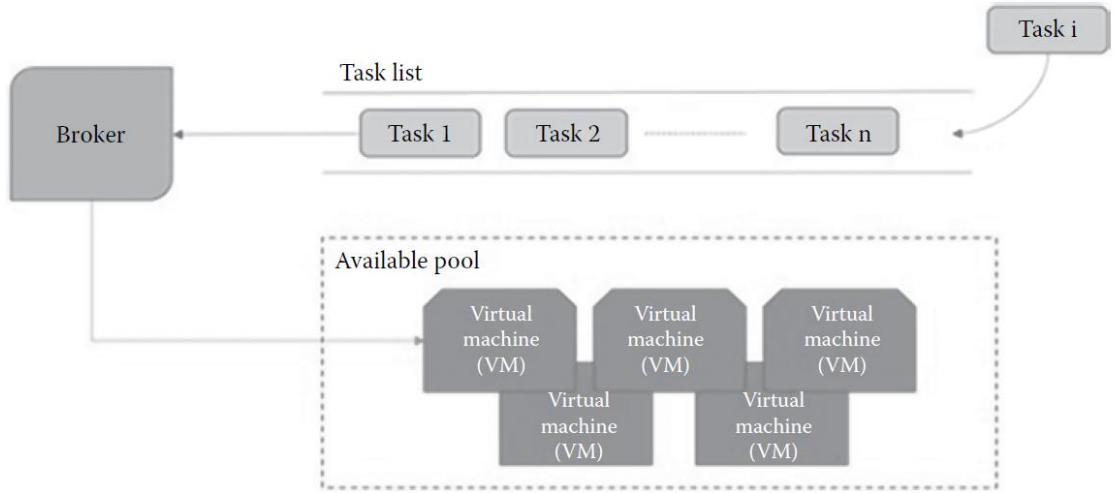

FIGURE 5.1
First-come-first-serve algorithm.

3.1.1.1 First-Come-First-Serve Algorithm

The broker selects first task and scans through available pool of VMs that can execute this task. Algorithm does not validate whether VMs selected are the most efficient to perform the execution.

3.1.1 List Scheduling (2)

3.1.1.2 Round-Robin Algorithm

- Round-robin algorithm manages task execution in a *First-In-First-Out (FIFO)* manner; however, each task is given a limited CPU time slice in a given VM (Salot, 2013). So, in the event of a task that is not completed during the time slice, the CPU is released for the next task waiting for execution in the queue. The preempted task is then placed at the back of the ready list.

3.1.1.3 Min–Min and Max–Min Algorithm

- Min–Min algorithm calculates execution time for tasks and creates a list of tasks in an increasing order of their execution time. Thus, small tasks are chosen to be executed first. The broker allocates available VMs for tasks in that order.
- However, the drawback is that the larger tasks which may be critical are executed at last in the sequence.
- Max-Min – largest first

3.1.2 Directed Acyclic Graph Scheduling

- One of the basic assumptions in list scheduling is that there are no interdependence tasks. Each task that arrives in the queue is independent and exclusive to any other task in the queue. DAG scheduling can handle such interdependence. DAG is a generic model for a parallel program that consists of a set of processes (nodes) and dependencies (edges).
- Each VM has its own memory and also connected to a shared memory for storing common messages among the tasks.

DAG

Assuming that each task takes timeslot of T1 (where T1 = T2 = T3 = …), Schedule 1 represents the optimal timelines for completing all tasks with dependencies intact (Task b needs to complete before Task c).
Schedule 2 depicts sequential pick of network (Route: Start → a → End, then Route: Start → b → d → End and at least route: Start → c → End). This schedule may not be optimal compared to Schedule 1, but has that constraint that needs to be complied with.

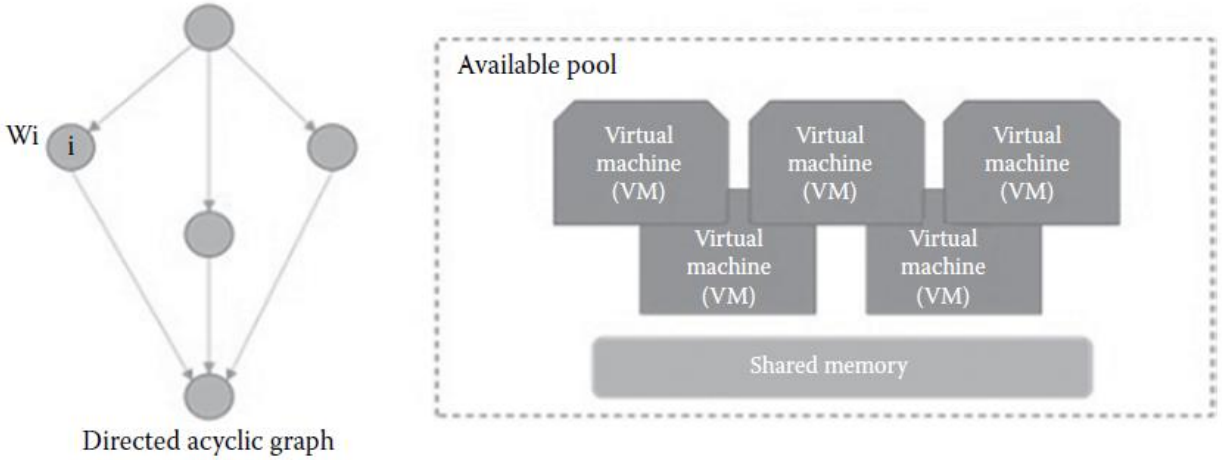

FIGURE 5.3
DAG and available pool of VMs.

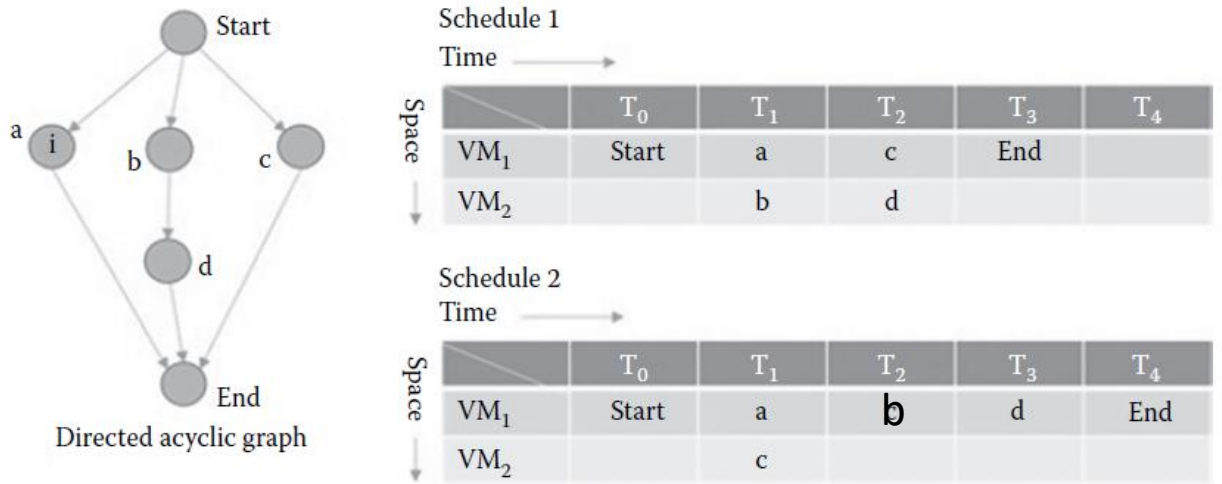

FIGURE 5.4
DAG scheduling scenarios using two VMs.

3.2 Data Storage Technology and Algorithms in Cloud

3.2.1 Programming Paradigm for Cloud Applications

- *5.3.2.1.1 Parallel Virtual Machine*
- Parallel Virtual Machine (PVM) is a programming model that provides framework for connecting computing devices in a network.
- The design provides abstraction of several heterogeneous machines to form a single computing entity.
- The underlying technique for PVM is message-passing model that exploits distributed computing across heterogeneous computing devices, including Massively Parallel Processing (MPP).
- This model handles message routing, data conversion, and task scheduling among the computing devices. The PVM library is portable and available as open source in the Internet.
- The user can construct one or more sequential programs in different programming languages such as C, C++, or Fortran 77, containing calls to PVM library.

3.2 Data Storage Technology and Algorithms in Cloud

3.2.1 Programming Paradigm for Cloud Applications

3.2.1.1 Parallel Virtual Machine

- Each program represents a task for a given application.
- These programs are compiled in a host pool of computing machines, and the compiled objects are kept in a location that is accessible from machines in the host pool.
- In order to execute an application, the tasks are triggered from any machine within the host pool.
- Machines are usually using *Single Program Multiple Data (SPMD)* type of computing model.
- This model then starts other PVM tasks that result in a collection of active tasks. These tasks are computed on different machines and can exchange messages with each other to solve the problem.
- This model can be used at *Infrastructure as a Service (IaaS)* layer to utilize resources and perform data-intensive computations.

3.2.1.1 Parallel Virtual Machine (2)

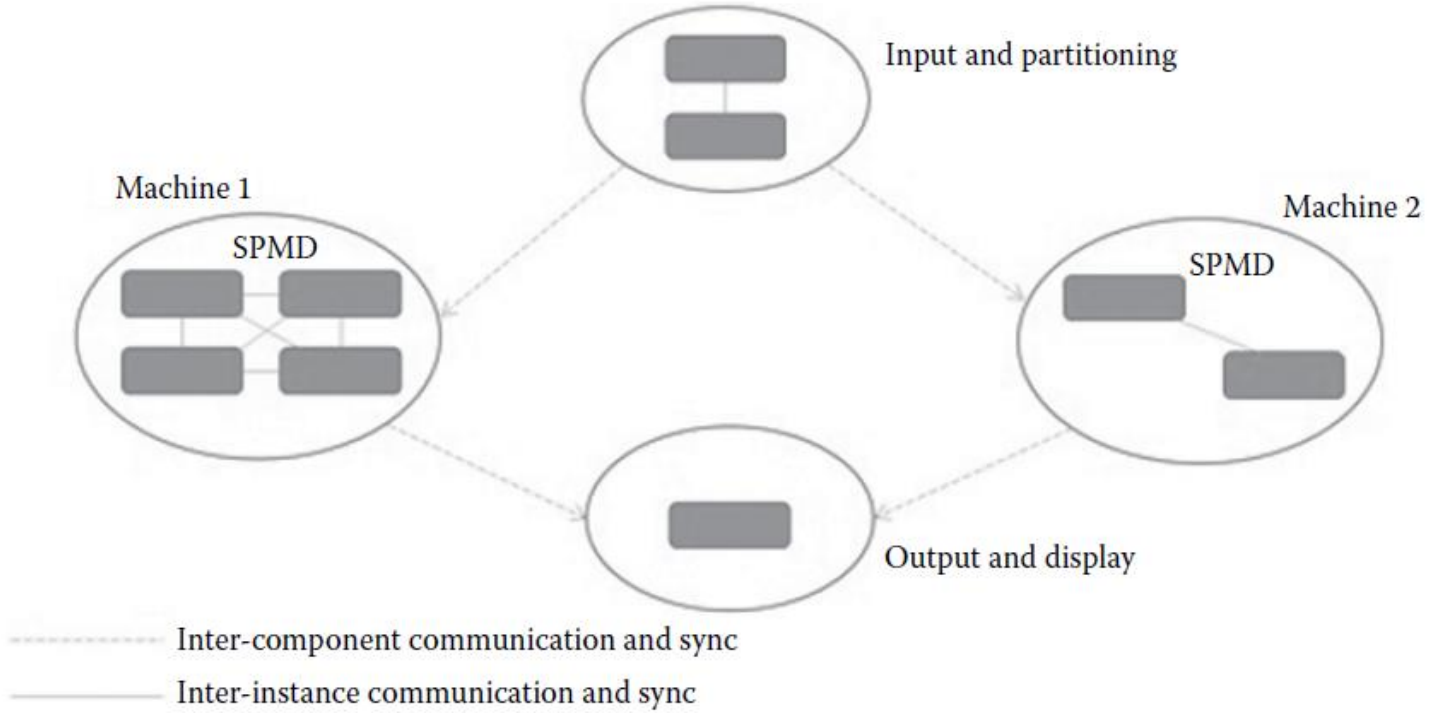

FIGURE 5.6
PVM programming model.

3.2.1.2 Message-Passing Interface

- MPI provides interface for communicating messages among a cluster of computing machines.
- The MPI interface specifies the functionality of high-level communication routines.
- Functions give access to a low-level implementation that takes care of sockets, buffering, data copying, message routing, etc.
- MPI applications are fairly portable and an efficient way of utilizing computing infrastructure.
- Some of the open MPI implementations are MPICH and OpenMPI.

MPI (2)

Step 1: Include the implementation-specific header file

#include inserts basic definitions and types

Step 2: Initialize communications –

MPI_Init initializes the MPI environment

MPI_Comm_size returns the number of processes

MPI_Comm_rank returns this process's number (rank)

Step 3: Communicate to share data between processes –

MPI_Send sends a message

MPI_Recv receives a message

Step 4: Exit from the message-passing system –

MPI_Finalize

FIGURE 5.7
An algorithm using MPI interface calls.

3.2.1.3 Map Reduce

- This programming model is currently adopted for processing large sets of data.
- The fundamental tenets of this model are map and reduce functions. The function of Map is to generate a set of intermediate key and value pairs, while Reduce function merges all intermediate values with same key.
- The model provides an abstract view of flow of data and control, and the implementation of all data flow steps such as data partitioning, mapping, synchronization, communication, and scheduling is made transparent to the users.
- User applications can use these two functions to manipulate the data flow. Data-intensive programs that are based on this model, can be executed *PaaS*.
- Hadoop, an open implementation by Apache, is based on Map Reduce model that has been used by many companies such as AOL, Amazon, Facebook, Yahoo, and New York Times for running their business application.

3.2.2 Cloud Storage Architecture

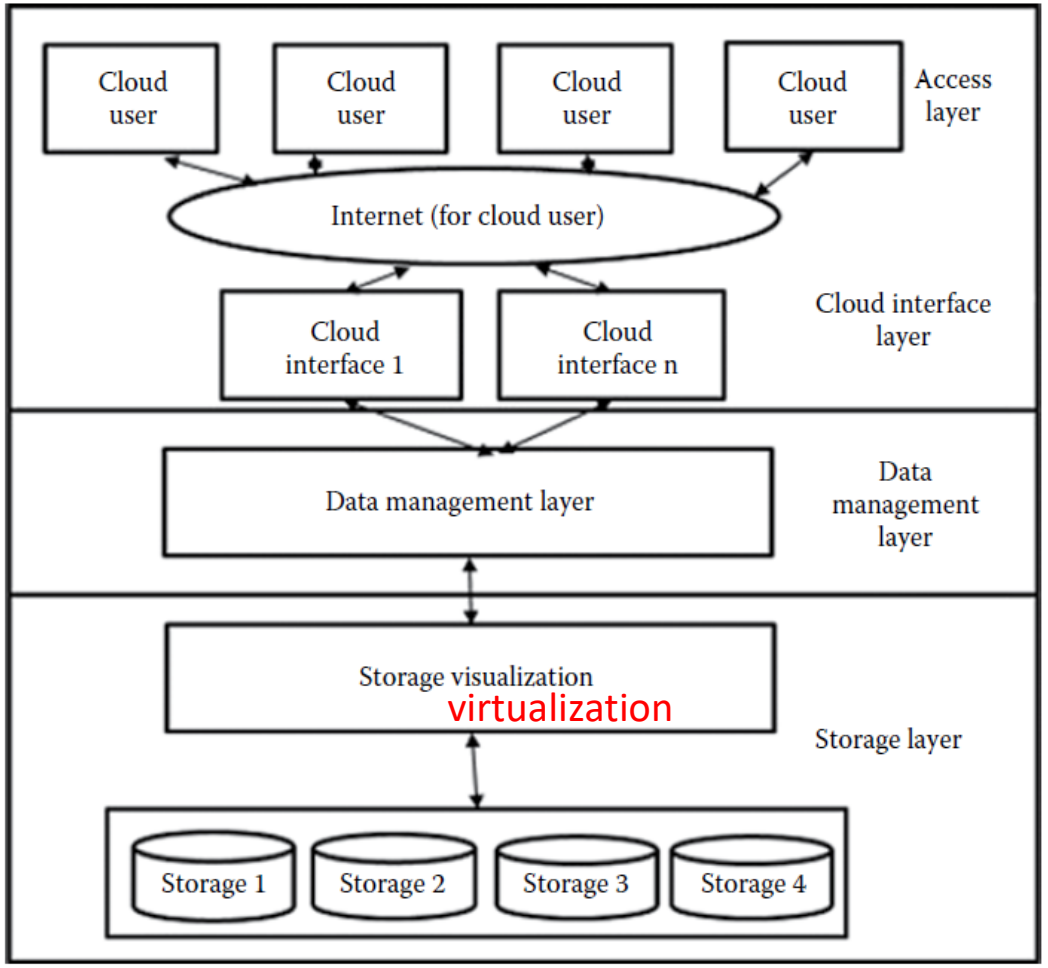

FIGURE 5.9
Generalized architecture of cloud storage. (Adapted from Elzeiny, A., Elfetouh, A. A., and Riad, A. 2013. *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, 2(4), 342–349.)

3.2.2.1 Access Layer

- Access layer is the primary interaction point for applications and cloud storage users which is accessible to the authorized user through a standard application interface from anywhere across the globe.
- This layer is also called as data service layer. There are various cloud storage operating units which facilitate various access types to the cloud storage systems, for example, web access.

3.2.2.2 Cloud Interface Layer

- Cloud storage providers provide a software layer to create the connection between cloud storage service and cloud user via Internet.
- User authorization and authentication are executed in this layer.
- This is also known as application interface layer or data service layer.
- This layer can be flexibly expanded and directly user-oriented.
- There are various application interfaces created to provide various services like data storage, video surveillance, network drivers, and multiuser sharing depending on various types of businesses and user requirements.

3.2.2.3 Data Management Layer

- Managing a specific cloud client is the principal job for its software layer.
- This layer is responsible for producing an integrated view of public administration required for the upper cloud storage system.
- This integrated public data management interface ensures the correct connectivity between storage layer and application interface layer.
- Here, clustered system, distributed file system, and grid computing technology are used to provide stronger storage devices, which is based on the implementation of coordinating various storage devices in cloud storage system.

3.2.2.3 Data Management Layer

Data management includes the following activities:

- Storing data
- Distribution of content across storing locality
- Partitioning of data
- Synchronizing
- Consistency maintenance
- Reproduction
- Controlling data flow over network
- Backup
- Recovery of data
- Large number of user handling
- Metadata and catalog maintenance

3.2.2.4 Storage Layer

- The most preliminary part of cloud data storage is Storage layer. Virtualization and Basic storage are the two main sections of this layer.

3.2.2.4.1 Basic Storage Basic storage consists of database servers and heterogeneous storage devices across various locations, where these servers or storage devices are connected through wide area network, Internet.

3.2.2.4.2 Storage Virtualization Storage virtualization is a technique for storing various types of storage devices and managing as an integrated single unit so that a single view of storage appears.

It maps from heterogeneous storage devices, which are also distributed in nature, to a specific storage location, continuous in nature, and also responsible for creating a shared dynamic platform.

Virtualization techniques: storage tiering and thin provisioning.

Thin provisioning: Provisioning as per the requirement.

- An application is allocated storage from a common pool based on the need. With this thin provisioning, usually logical storage is allocated by storage administrator to an application, but in reality, the system releases the actual physical capacity only at the time of requirement. This feature provides an optimal utilization of storage by reducing the amount of allocated but unused physical storage.

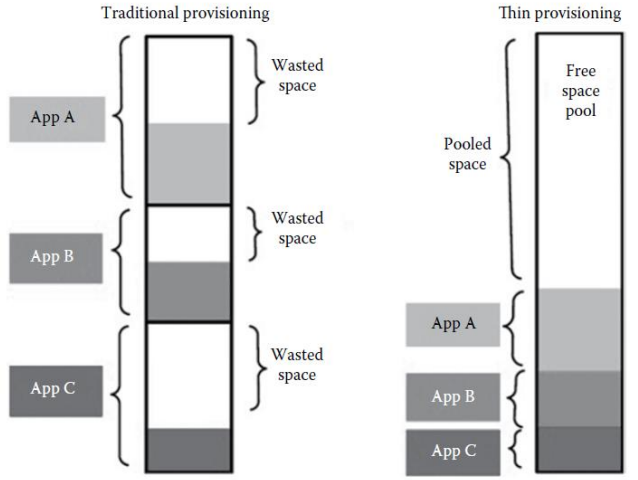

FIGURE 5.11
Traditional provisioning versus thin provisioning. (Adapted from Thin Provisioning by NetApp Community, last edited on 2016. Available at http://community.netapp.com/t5/Technology/Space-Reclamation-If-Being-Thin-is-Attractive-Then-Thin-Provisioning-Makes-for/ba-p/82836.)

Virtualization techniques: storage tiering and thin provisioning.

Storage tiering: Current techniques should be in place that enables to store right data in right location and make them accessible in the correct time by the organizations.

- Hierarchy of storage types can be established where data can be stored in the hierarchy based on their performance, availability, and recovery requirements.
- As an example, drives with high performance can be configured as tier1 storage for keeping frequently accessed data, which will improve performance.
- On the other hand, low-cost drives can be configured as tier2 for residing data that are not accessed frequently. Here, tier1 space is freed up, which increases the capacity of high-performance drives and reduces the cost of storage. These data movement happens based on few predefined properties, like file type, access frequency, and performance.

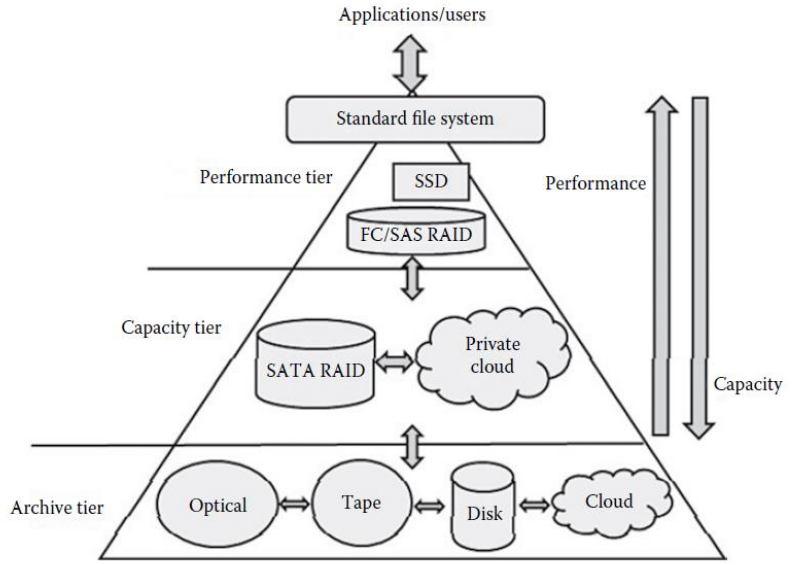

FIGURE 5.12
Automated tiering. (Adapted from Automated Tiering by euroson—Archival and Storage Solutions. Available at http://www.euroson.com/products/point-software/point-storage-manager/.)

Virtualization benefits

- To provide cost-effective ways for eliminating single point of failure in the storage area network.
- Improvement of service quality by applying proper management and process improvement technique in real time.
- Disaster recovery and data archival can be achieved by providing cost-effective solutions.
- Utilization, scalability, and flexibility improvement across the storage area network.

3.3 Distributed Data Storage Algorithms and Efficient Data Storage Strategies

Data storage in cloud platform enables user to store and retrieve data in a remote and distributed fashion without having the burden of hardware and software in their local machines along with the availability of on-demand, high-quality cloud applications.

There are a few top advantages of cloud data storage, which include the following:

- I. *Flexibility*: Users are allowed to drag and drop files between the cloud storage and their local storage.
- II. *Accessibility*: The stored files in the cloud can be accessed from anywhere across the globe via Internet.
- III. *Disaster Recovery*: As the files are stored in the remote location, they are safe in case of any disaster that causes damage to local files that consists of critical business data.
- IV. *Cost savings*: As storage can be paid based on usage, expenditure can be reduced while data volume is low.

3.3.1 Distributed Data Storage Algorithms

- Cloud-based storage has few inherent vulnerabilities. The biggest challenges in the cloud storage include data security, reliability, and data access efficiency.

3.3.1.1 Data Security

- Data security is a common threat in any technology domain, but it is more vulnerable to the cloud infrastructure. Various security issues may arise in the areas like external data storage, dependency on the “public” Internet and absence of control, resource sharing and integration with internal security. Encryption technique is the most common and effective way to protect sensitive data from the past.

3.3.1.2 Reliability

- Reliability is broadly a function of reliability of four individual components: the hardware and software facilities offered by providers, the provider's personnel, and connectivity to the subscribed services and the consumer's personnel).
- Reliable data movement ensures successful transfer of data from source to target without any malicious attacks in the middle way.

3.3.1.3 Data Access Efficiency in Cloud

- The biggest advantage of cloud storage lies in its flexibility, which allows users to access data at any time.
- Here, in the cloud system, user requirements are analyzed automatically by storage management system. It also finds and transforms data which helps the users to a great extent.
- The Huffman coding is an algorithm that can be used to better data access performance in cloud environment.

REDUCE THE DATA!

3.3.2.3 Data Reduction Techniques

- "The real savings in energy efficiency comes in managing the data," so automated data retention strategies should be in place to safely eliminate or copy data to tape at the end of their preset retention periods. Data classification can play an effective role in wiping out unnecessary data.
- Basic steps for effective data management and reduction:
  - *Gather clear understanding on capacity*: Data storage environment needs to be thoroughly profiled so that appropriate places can be identified to increase the efficiencies. This will also help later during the tiering process.
  - *Remove unwanted files*: Several duplicate files or irrelevant files stored across the server should be identified and deleted in a regular basis for achieving better performance.
  - *Appropriated data tiering*: Organization's data should be tiered in a proper way.
  - *Watch the trends*: Adequate attention should be paid to the utility of the resources along with the storage environments. Proper techniques can be applied accordingly to cope with the changes.

3.3.2.4 Data Deduplication Techniques

- Deduplication is the technology that shrinks the data footprint by removing redundant data at the file or subfile level. This is the most common technology with backups and archives, but it increases acceptance with primary storage.

3.3.2.5 Data Compression Techniques

- Data compression reduces the size of data based on the algorithms. It can be done separately or along with data deduplication.

Performance management in cloud

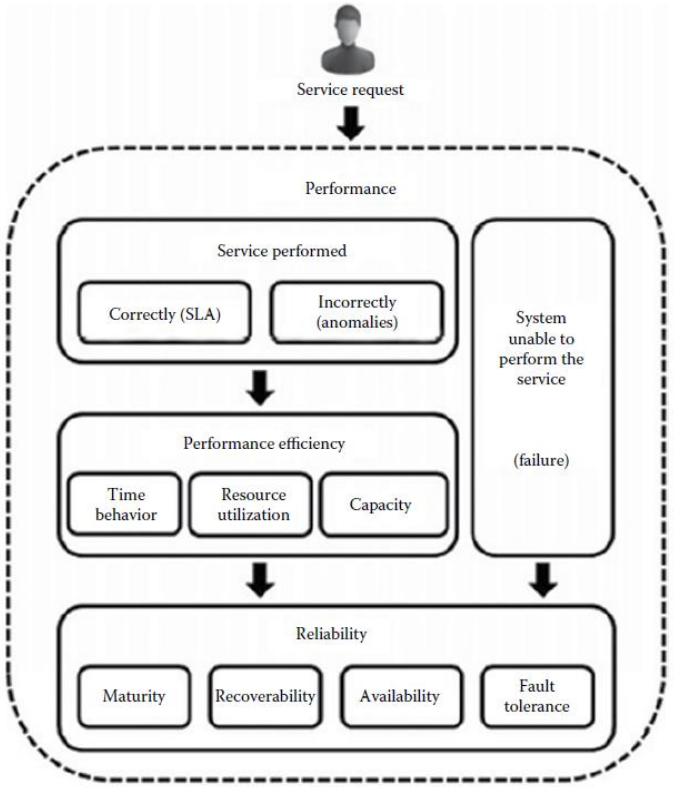

FIGURE 5.13
Performance management in cloud. (Adapted from Bautista, L., Abran, A., and April, A. 2012. *Journal of Software Engineering and Applications*, 5(2), 69–75.)

Performance management

- *Performance efficiency*: The amount of resources used under specific circumstances, which includes both the software and the hardware products.
- *Capacity*: Defines the degree at which requirement of a product is met.
- *Resource utilization*: Describes the type and quantity of resources for any particular product or system.
- *Time behavior*: It includes the reply, processing time, and throughput rate of a product to meet the requirements.
- *Reliability*: It measures whether a system can perform specified functions under particular conditions for a certain period of time.
- *Maturity*: It measures whether the reliability requirements are met under normal operation or not.
- *Availability*: The degree at which a system is functional and accessible at the time of requirement.
- *Fault tolerance*: It measures whether the system is operational if any software or hardware failure occurs.
- *Recoverability*: The degree at which a system can recover data directly at the time of failure or interruption and can be restored to the expected condition.